sciendo

VOLUME 18, N° 2 2024
Journal of Automation, Mobile Robotics and Intelligent Systems

Effective Nonlinear Predictive and CTC-PID Control of Rigid Manipulators

Submitted: 24th October 2023; accepted: 14th December 2023

Piotr Tatjewski

DOI: 10.14313/JAMRIS/2-2024/8

Abstract:
Effective nonlinear control of manipulators with dynamically coupled arms, like those with direct drives, is the subject of the paper. Model-based predictive control (MPC) algorithms with nonlinear state-space models and most recent disturbance attenuation technique are proposed. This technique makes controller design and online calculations simpler, avoiding necessity of dynamic modeling of disturbances or resorting to additional techniques like SMC. The core of the paper are computationally effective MPC-NPL (Nonlinear Prediction and Linearization) algorithms, where computations at every sample are divided into two parts: prediction of initial trajectories using nonlinear model, then optimization using simplified linearized model. For a comparison, a known CTC-PID algorithm, which is also model-based, is considered. It is applied in standard form and also proposed in more advanced CTC-PID2dof version. For all algorithms a comprehensive comparative simulation study is performed, for a direct drive manipulator under disturbances. Additional contribution of the paper is investigation of influence of sampling period and of computational delay time on performance of the algorithms, which is practically important when using model-based algorithms with fast sampling.

Keywords: *manipulator control, nonlinear control, model predictive control, CTC-PID control, fast sampling, delays*

1. Introduction

Robotic manipulators have been broadly utilized in various automated industrial applications. Controllers are needed for the manipulators, which assure adequate tracking performance during fast, safe and smooth motions. Robotic manipulators are complex, nonlinear mechatronic systems. Thus, the controllers must perform nonlinear and multivariable motion tasks. There are many solutions to this problem presented for years in the literature, starting from classical multiloop PID control structures. This solution is still popular in practice, especially for systems with gears which can effectively damp coupling effects between individual links. It is not the aim of this paper to deliver a review of structures and algorithms of manipulator control, with full list of related references. We are lucky that a review paper has been very recently published on the subject [1], the interested reader is referred there.

An earlier comprehensive review paper can be here also recommended [2]. Textbooks on manipulator control, including standard PD/PID control and more elaborate PID structures with gravity compensation and with computed torque control (CTC), are also worth mentioning [3, 4].

Model-based predictive control (MPC) is now an established advanced control technology, represented by numerous algorithms and software packages applied successfully in industrial practice, especially to process control, see, e.g., [5–13]. MPC algorithms have been successfully applied first to industrial multivariable systems with strong interactions, active constraints, difficult dynamics, where classical PID control could not provide satisfactory performance. MPC is a model-based technology needing more computing power; thus, it was first applied to processes with slow dynamics. Applications to faster processes were then developed, on the one hand, by application of more powerful and faster computers, and on the other hand, by development of suboptimal but simpler and faster versions of algorithms. The latter development took place when constructing MPC algorithms with nonlinear process models. Then simpler, easier computable process model versions or approximations were used, like linear parameter-varying (LPV), fuzzy or neural network models, and nonlinear models of special structure like Wiener models, see, e.g., [8, 9, 14–17].

Robotic manipulators are nonlinear systems performing fast motions, thus operating with short sampling periods. Therefore, direct applications of MPC with nonlinear model and online optimization have been difficult and were reported in the literature rarely and only recently, with specific developments to make optimization more efficient, see [18, 19]. On the other hand, the nature of the manipulator control problem, a multivariable one with possible strong interactions and active constraints seems perfectly suited for application of MPC. Therefore, there were trials reported in recent years to develop simpler MPC-based manipulator control. The main difficulty, online nonlinear optimization, was usually replaced by a quadratic one using linear approximations, or by problems with LMI (linear matrix inequalities) constraints. One approach was to use a LPV representation of the nonlinear model, see, [20, 21]. The other and more often met approach was to use an online linearization.

2024 © Piotr Tatjewski. This is an open access article licensed under the Creative Commons Attribution-Attribution 4.0 International (CC BY 4.0)
(http://creativecommons.org/licenses/by-nc-nd/4.0/)

Wilson et.al. [22] applied DMC algorithm with step response model recalculated at every sampling instant by simulation of the nonlinear model. Linearization of the model at current point was applied, e.g., in [9, 21, 23]. The linear model was then used both to output prediction and quadratic optimization at every sample (the MPC-NSL method, see [9]). To overcome difficulties due to nonlinearity, the approach with MPC applied to the simplified linear system after feedback linearization by CTC (computed torque control) has been proposed, see [24–27]. However, it seems that application of MPC to extremely simple linear system after CTC linearization is not a good solution. MPC itself is a powerful technique capable to cope with nonlinearities, equipped also with effective mechanisms of disturbance attenuation. Its potential cannot be well utilized when feedback linearization is applied with not adequately accurate model, as MPC cannot then utilize structural properties of the original manipulator model.

The problem to attenuate disturbances, both these stemming from model inaccuracies (parameter uncertainty, unmodeled dynamics) and these from external influences, is a vital one in control system design. To cope with it, a combination of MPC with PID [22] or with SMC (sliding mode control) has been proposed, see [24, 25, 27]. However, this led to unnecessarily overcomplicated, multiloop control systems. The reason for that was that only recently an efficient technique of disturbance modeling and attenuation was proposed for MPC with state-space models, with first applications to process control systems. It is especially effective for cases with measured state as it removes then the need to apply dynamic observers/filters of disturbances, see [28–31]. The two latter papers consider both cases, with measured and unmeasured (observed, filtered) state, and a significant generalization from measured to unmeasured state. The case with measured state is often met in manipulator control, with the state consisting of both positions and velocities.

The aim of this paper is to present an application of state-of-the-art realizations of nonlinear MPC algorithms to manipulator control, not resorting to SMC or other additional techniques to attenuate disturbances or to feedback linearization by CTC. In Section 2, the appropriate MPC algorithms will be presented. First, the most general MPC-NO algorithm (MPC with Nonlinear Optimization) using a nonlinear manipulator model will be briefly described, further used as a benchmark. Then, much more effective MPC algorithms with nonlinear prediction and linearization (MPC-NPL) will be presented, which are the core of the paper. The first one is with appropriately organized constrained quadratic programming instead of nonlinear optimization. The next, computationally most effective, will be derived as an analytical (explicit) version of the previous one. All the algorithms will be using the disturbance attenuation technique as proposed in [29, 30]. Organization of the calculations within the MPC-NPL algorithms leading to shortest execution times will be also discussed.

In Section 3, results of a comprehensive comparative simulation study of applications of all the algorithms to control a direct drive manipulator will be reported, for different reference trajectories and different disturbances. These include external disturbing torque and substantial differences between manipulator and its model in the feedback loop, both in parameters and in unmodelled dynamics. The results will be also compared with those for the CTC-PID algorithm, in known standard form and in enhanced CTC-PID2dof structure. Investigations of the influence of sampling period and of a delay caused by the time of computations on algorithms performance will be an additional contribution of the paper.

The paper is a significantly extended and English version of the paper [32] published recently in Polish. Except for minor improvements, main extensions involve, firstly, a new concept of computational structures of the MPC-NPL algorithms leading to shortest execution times. Further significant extension is the design and investigation of effective versions of both MPC and CTC-PID algorithms for manipulator control for piecewise constant reference trajectories. This includes MPC with additional internal reference trajectories and CTC with multiloop PID in 2dof structure. All the described extensions are validated by application to control of a direct drive manipulator mentioned above. Conclusions will be the last part of the paper.

2. Model Predictive Control Algorithms

2.1. Models of the Manipulator

We consider the following well-known nonlinear continuous-time dynamic model of a n-dof rigid body robot manipulator [1–4],

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + F(\dot{q}) + g(q) = u, \qquad (1)$$

where q, $\dot{q}$, $\ddot{q} \in R^n$ are vectors of joint positions, velocities and accelerations, respectively, $M(q)$ is the inertia matrix, $C(q,\dot{q})$ is the matrix of centripetal and Coriolis forces, $F(\dot{q})$ is the function representing friction, $g(q)$ is the gravity vector, and u is the control input vector (motor torques).

Define the state vector $x^T = [\, q^T \;\; \dot{q}^T \,] \in R^{2n}$. Then, (1) can be written in standard form

$$\dot{x} = \begin{bmatrix} \dot{q} \\ -M(q)^{-1}[C(q,\dot{q})\dot{q} + F(\dot{q}) + g(q)] + M(q)^{-1}u \end{bmatrix} \qquad (2)$$

Using the Euler discretization scheme with integration step T_c, we get the discrete-time model:

$$x(k+1) = x(k) + \begin{bmatrix} T_c\dot{q}(k) \\ -T_c M(q(k))^{-1}[C(x(k))\dot{q}(k) + \\ +F(\dot{q}(k)) + g(q(k)) - u(k)] \end{bmatrix}, \qquad (3)$$

$$y(k) = q(k). \qquad (4)$$

$$\min_{u(k|k),\ldots,u(k+N_u-1|k)} \left\{ J(k) = \sum_{p=1}^{N} \left\| y^{ref}(k+p|k) - y(k+p|k) \right\|_{\Psi}^{2} + \; + \sum_{p=0}^{N_u-1} \left\| u(k+p|k) - u(k+p-1|k) \right\|_{\Lambda}^{2} \right\}$$

$$\begin{aligned} \text{subject to}: \; & - \text{prediction equations calculating } y(k+p|k) \text{ using the model} \\ & \quad \text{for given values } u(k|k), \ldots, u(k+N-1|k),\ p = 1, \ldots, N, \\ & - u_{min} \le u(k+p|k) \le u_{max},\ p = 0, \ldots, N_u - 1, \\ & - \triangle u_{max} \le u(k+p|k) - u(k+p-1|k) \le \triangle u_{max},\ p = 0, \ldots, N_u - 1, \\ & - y_{min} \le y(k+p|k) \le y_{max},\ p = 1, \ldots, N \end{aligned} \tag{7}$$

A simplified, continuous and differentiable model with viscous friction only, $F(\dot{q}) = F_v \dot{q}$ with diagonal matrix F_v, will be used for MPC design, treating static (Coriolis) friction as unmodelled dynamics. This model will be further written in standard short form

$$x(k+1) = f(x(k), u(k)), \tag{5}$$

$$y(k) = \mathbf{C}x(k). \tag{6}$$

2.2. MPC-NO Algorithm (with Nonlinear Optimization)

Denote by N the number of sampling periods T_c defining the prediction horizon and by N_u the number of sampling periods defining the control horizon. We define the dynamic optimization problem of MPC-NO controller as presented in Eq. (7), where $y(k+p|k)$, $u(k+p|k)$, etc., denote values of the manipulator outputs and control inputs, respectively, predicted for the sample $k+p$ at the current sample k, see, e.g., [9, 30]. For $p = 0$, we have $u(k+0-1|k) = u(k-1)$, which is the manipulator control input calculated at the previous sampling instant. For $p \ge N_u$, we set $u(k+p|k) = u(k+N_u-1|k)$. Matrices Ψ, Λ are diagonal matrices of positive weighting coefficients and $||x||_{\Psi} \triangleq \sqrt{x^T \Psi x}$. The control trajectory $U(k)$ is the decision vector of (7),

$$U(k)^T = \left[u(k|k)^T \; u(k+1|k)^T \; \cdots \; u(k+N_u-1|k)^T \right] \tag{8}$$

For formulation of output predictions over the prediction horizon N, the applied disturbance attenuation technique is crucial. In [28], "constant state disturbance model" was presented for MPC with linear state-space models, with the proof showing it assures offset-free control for asymptotically constant disturbances (including modeling errors). In [29, 30] generalizations to nonlinear systems were given. This constant state disturbance model is as follows,

$$v(k) = x(k) - x(k|k-1) = x(k) - f(x(k-1), u(k-1)). \tag{9}$$

This means that $v(k)$ is the difference between the state $x(k)$ measured at current sample k and the state $x(k|k-1)$ predicted for this sample at previous sample $k-1$. It is "constant" as the same value $v(k)$ is assumed over whole prediction horizon, i.e.,

$$v(k+1|k) = v(k+2|k) = \cdots = v(k+N-1|k) = v(k). \tag{10}$$

Having defined the disturbance model, state and then output prediction equations can be formulated, for any given control input trajectory (8):

$$\begin{aligned} x(k+1|k) &= f(x(k), u(k|k)) + v(k), \\ x(k+2|k) &= f(x(k+1|k), u(k+1|k)) + v(k), \\ &\vdots \\ x(k+N|k) &= f(x(k+N-1|k), u(k+N-1|k)) + v(k), \end{aligned} \tag{11}$$

where $u(k+p|k) = u(k+N_u-1|k)$, $p = N_u, \ldots, N-1$,

$$y(k+p|k) = \mathbf{C}x(k+p|k),\ p = 1, \ldots, N. \tag{12}$$

With prediction equations (11)-(12), MPC optimization problem (7) is fully defined and can be solved, at each sampling instant. The first element $\hat{u}(k|k)$ of the obtained optimal control trajectory $\hat{U}(k)$ is then sent to manipulator actuators as the current control signal $u(k)$. At the next sampling instant new measurements are available and the whole MPC algorithm is repeated (receding horizon technique).

Subsequent MPC optimization problems (7) usually differ only slightly at consecutive samples, with changes in a few parameters: new state measurement $x(k)$ and last control values $u(k-1)$. In such cases, the technique of "warm start" may be efficient, i.e., using previous optimal control trajectory $\hat{U}(k-1)$ as the basis for new initial control trajectory $U^0(k)$. Due to the receding horizon, this consists in modifying the $\hat{U}(k-1)$ by omitting the first subvector $\hat{u}(k-1|k-1)$ and repeating the last one. The warm start should be efficient for longer control horizons N_u. It is usually not the case of manipulator control, when fast sampling and reduced computational load are required.

The MPC-NO algorithm will be treated further in the paper as an optimal pattern to be compared with faster but suboptimal MPC algorithms. However, due to development of still more powerful and cheaper microcontrollers and improvements in nonlinear optimization procedures, the area of possible applications of algorithms with nonlinear optimization is widening, see, e.g., [18, 33].

2.3. MPC-NPL Algorithms (with Nonlinear Prediction and Linearization)

A straightforward way to simplify the MPC-NO algorithm is to construct an adaptive algorithm that, at each sampling instant, linearizes the nonlinear model and then uses the standard MPC algorithm with the linear model for prediction and optimization. It is known as MPC-NSL technique (Nonlinear with Successive Linearizations), see, e.g., [6, 9, 21]. However, this algorithm may not be successful, especially for systems with stronger nonlinearities. The computationally comparable and more widely applicable approach is the algorithm that, at each sampling instant, firstly performs prediction of the initial output trajectory using the full nonlinear model. Then, in the second phase, it performs linearization and applies optimization using the incremental linear model, to improve the initially computed trajectory. Such algorithm will be denoted by MPC-NPL (MPC with Nonlinear Prediction and Linearization), see, e.g., [9].

Subsequent steps of the MPC-NPL algorithm, at k-th sampling instant:

1) The manipulator state $x(k)$ is measured (or estimated), $v(k)$ is calculated according to (9).
2) Initial trajectory of control inputs $U^0(k)$ is determined over the prediction horizon,

$$U^0(k)^T = [\, u^0(k|k)^T \; u^0(k+1|k)^T \cdots \; u^0(k+N-1|k)^T \,].$$

Using $U^0(k)$, the initial trajectory of states $X^0(k)$ (consisting of $x^0(k+p|k), p = 1, \ldots, N$) and then the initial trajectory of outputs $Y^0(k)$ (consisting of $y^0(k+p|k)$, $p = 1, \ldots, N$) are calculated using the nonlinear model:

$$x^0(k+p|k) = f(x^0(k+p-1|k), u^0(k+p-1|k)) + v(k), \quad (13)$$

$$y^0(k+p|k) = \mathbf{C} x^0(k+p|k), \; p = 1, \ldots, N, \quad (14)$$

where $x^0(k|k) = x(k)$ is the measured state.

3) The state equation is linearized at current point $(x(k), u^0(k|k))$. Matrix $\mathbf{M}(k)$ describing linear relation between increments of control (over initial control trajectory) and corresponding increments of outputs (over initial output trajectory) is calculated,

$$\triangle Y(k) = \mathbf{M}(k) \cdot \triangle U(k), \quad (15)$$

where

$$\triangle U(k)^T = \left[\triangle u(k|k)^T \; \triangle u(k+1|k)^T \; \cdots \; \cdots \; \triangle u(k+N_u-1|k)^T \right], \quad (16)$$

$$\triangle Y(k)^T = \left[\triangle y(k+1|k)^T \; \triangle y(k+2|k)^T \; \cdots \; \cdots \; \triangle y(k+N|k)^T \right], \quad (17)$$

where

$$\triangle u(k|k) = u(k|k) - u(k-1),$$
$$\triangle u(k+1|k) = u(k+1|k) - u(k|k),$$
etc.

4) The MPC-NPL quadratic optimization problem (23) with the linearized model (15) is solved, with the solution $\widehat{\triangle u}(k+p|k), p = 0, \ldots, N_u - 1$.
5) The (sub)optimal control trajectory is calculated:

$$\widehat{u}(k+p|k) = u^0(k+p|k) + \sum_{j=0}^{p} \widehat{\triangle u}(k+j|k), \quad p = 0, \ldots, N_u - 1,$$
$$\widehat{u}(k+p|k) = \widehat{u}(k+N_u-1|k), \quad p = N_u, \ldots, N-1. \quad (18)$$

6) First subvector of the optimal control trajectory, $\widehat{u}(k|k)$, is sent to the actuators as the current controller output (manipulator control input) $u(k)$.

Initial control trajectory $U^0(k)$ can be either a constant one, consisting of last manipulator control input $u(k-1)$, i.e., $u^0(k+p|k) = u(k-1), p = 0, \ldots, N-1$, or the trajectory according to "warm start" technique, i.e., $u^0(k+p|k) = \widehat{u}(k+p|k-1), p = 0, \ldots, N-2, u^0(k+N-1|k) = \widehat{u}(k+N-2|k-1)$. The first is recommended for very short control horizon ($N_u \leq 2$) and small T_c, a case typical for fast dynamics. It is the case of manipulator control; thus, further presentation will be with this trajectory. The latter one may increase efficiency for longer N_u, but needs a reformulation of $J(k)$ in (23). This will be omitted, due to the given reason and to shorten the length of the paper.

At point 3, an incremental linear state-space model is obtained after linearization

$$x(k+1) = \mathbf{A}(k)x(k) + \mathbf{B}(k)u(k), \quad (19)$$

The manipulator model (3) is linear with respect to $u(k)$, hence linearization is not needed to get $\mathbf{B}(k)$, $\mathbf{B}(k) = T_c M(q(k))^{-1}$. To get $\mathbf{A}(k)$, partial derivatives must be calculated for the last n state equations in (3) only, as shown in Eq. (20), where $\frac{\partial}{\partial x(k)}[\cdot]$ denotes matrix of partial derivatives of the function in square brackets (the Jacobi matrix). After linearization, matrices $\mathbf{A}(k)$, $\mathbf{B}(k)$ and $\mathbf{C}$ are used to calculate the dynamic matrix $\mathbf{M}(k)$, which has the structure

$$\mathbf{M}(k) = \begin{bmatrix} \mathbf{M}_1(k) & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{M}_2(k) & \mathbf{M}_1(k) & \cdots & \mathbf{0} \\ \mathbf{M}_3(k) & \mathbf{M}_2(k) & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{M}_{N_u}(k) & \mathbf{M}_{N_u-1}(k) & \cdots & \mathbf{M}_1(k) \\ \mathbf{M}_{N_u+1}(k) & \mathbf{M}_{N_u}(k) & \cdots & \mathbf{M}_2(k) \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{M}_N(k) & \mathbf{M}_{N-1}(k) & \cdots & \mathbf{M}_{N-N_u+1}(k) \end{bmatrix}, \quad (21)$$

where

$$\mathbf{M}_i(k) = \mathbf{C}\,[\mathbf{I} + \mathbf{A}(k) + \mathbf{A}(k)^2 + \cdots + \mathbf{A}(k)^{i-1}]\,\mathbf{B}(k), \quad (22)$$

with $\mathbf{M}_1(k) = \mathbf{C}\,\mathbf{B}(k)$. The structure (21)-(22) is standard for MPC algorithms with linear state-space models, see, e.g., [9, 34].

$$\mathbf{A}(k) = \begin{bmatrix} \mathbf{0}_n & \mathbf{I}_n \\ \frac{\partial}{\partial x(k)} \left[-T_c M(q(k))^{-1} \left(\left(C(x(k)) + F_v \right) \dot{q}(k) + g(q(k)) \right) \right]_{n \times 2n} \end{bmatrix} \tag{20}$$

$$\min_{\Delta U(k)} \left\{ J(k) = \sum_{p=1}^{N} \left\| y^{ref}(k+p|k) - y^0(k+p|k) - \mathbf{M}(k)\Delta U(k) \right\|_{\Psi}^2 + \right.$$
$$\left. + \sum_{p=0}^{N_u - 1} \left\| \triangle u(k+p|k) \right\|_{\Lambda}^2 \right\}$$
$$\text{subject to : } -u_{min} \le u(k-1) + \textstyle\sum_{j=0}^{p} \triangle u(k+j|k) \le u_{max},\ p = 0, \dots, N_u - 1, \tag{23}$$
$$-\triangle u_{max} \le \triangle u(k+p|k) \le \triangle u_{max},\ p = 0, \dots, N_u - 1,$$
$$-y_{min} \le y^0(k+p|k) + \mathbf{M}(k)\Delta U(k) \le y_{max},\ p = 1, \dots, N,$$

At point 4., the quadratic programming (QP) problem shown in Eq. (23) is solved. It is a standard form of the MPC optimization problem with a linear model, with (16) as the vector of decision variables, see, e.g., [9].

The fundamental difference between (23) and a QP problem for MPC with a linear model only is that now the initial trajectory of the outputs, $y^0(k+p|k)$, $p = 1, \dots, N$, is calculated using the nonlinear model, not the linear one, and the dynamic matrix $\mathbf{M}(k)$ is subject to adaptation.

2.4. Analytical (Explicit) MPC-NPL Algorithm

To reduce further online computations, an analytical (explicit) version of the MPC-NPL algorithm has been developed. The idea is to find, first, the point minimizing the quadratic function $J(k)$ of (23) only, neglecting temporarily inequality constraints. Next, the calculated (unconstrained) values are trimmed to limits of inequality constraints. This adds additional suboptimality, but significantly reduces the computational load. Moreover, application examples have shown that such faster algorithm can be almost as effective as the one with constrained QP optimization for cases with limited activity of inequality constraints (for inactive constraints results are equivalent).

The function $J(k)$ in (23) is strictly convex, thus its minimum can be easily calculated solving linear equations of necessary optimality conditions:

$$[\mathbf{M}(k)^T \underline{\Psi} \mathbf{M}(k) + \underline{\Lambda}] \Delta U(k) = \mathbf{M}(k)^T \underline{\Psi}\, [Y^{ref}(k) - Y^0(k)], \tag{24}$$

where $\underline{\Psi} = \text{diag}\{\Psi, \dots, \Psi\}$, $\underline{\Lambda} = \text{diag}\{\Lambda, \dots, \Lambda\}$ are block-diagonal matrices, consisting of N blocks Ψ and of N_u blocks Λ, respectively, and

$$Y^{ref}(k) = \begin{bmatrix} y^{ref}(k+1|k) \\ y^{ref}(k+2|k) \\ \vdots \\ y^{ref}(k+N|k) \end{bmatrix}_{nN \times 1}, \tag{25}$$

$$Y^0(k) = \begin{bmatrix} y^0(k+1|k) \\ y^0(k+2|k) \\ \vdots \\ y^0(k+N|k) \end{bmatrix}_{nN \times 1}. \tag{26}$$

The solution $\widehat{\Delta U}(k)$ of (24) yields the unconstrained (sub)optimal trajectory of control increments (over the initial trajectory) on the control horizon:

$$\widehat{\Delta U}(k)^T = \left[\widehat{\triangle u}(k|k)^T \ \ \widehat{\triangle u}(k+1|k)^T \ \cdots \right.$$
$$\left. \cdots \ \widehat{\triangle u}(k+N_u-1|k)^T \right] \tag{27}$$

The first subvector $\widehat{\triangle u}(k) = \widehat{\triangle u}(k|k)$ is now trimmed to lower and upper limits of rate of change constraints, $-\triangle u_{max}$ and $\triangle u_{max}$, respectively, with the result $\overline{\triangle u}(k)$. Next, $\overline{u}(k) = u(k-1) + \overline{\triangle u}(k)$ is trimmed to lower and upper limits of amplitude constraints, u_{min} and u_{max}, with the resulting control signal $u(k)$ satisfying all constraints. This is shown schematically in Figure 1, which presents the structure of the algorithm. It will be called MPC-NPL *analytical algorithm* (MPC-NPLa). It is "analytical" because the solution $\widehat{\Delta U}(k)$ of (24) can be expressed by analytical formula

$$\widehat{\Delta U}(k) = [\mathbf{M}(k)^T \underline{\Psi} \mathbf{M}(k) + \underline{\Lambda}]^{-1} \mathbf{M}(k)^T \underline{\Psi} \cdot$$
$$\cdot [Y^{ref}(k) - Y^0(k)]. \tag{28}$$

However, solving set of linear equations (24) needs significantly less computations than application of the inverse matrix formula (28). Therefore, it is recommended when $\mathbf{M}(k)$ is recalculated at each sampling instant. However, if it is recalculated less often, then using (28) may be reasonable.

The algorithm described in the previous section will be denoted MPC-NPLn (numerical), to distinguished it from MPC-NPLa.

2.5. Efficient Computational Structures for MPC-NPL

Main computational load when performing MPC-NPL algorithms is due to:

1. Calculation of the nonlinear model (13) N times to get the initial output trajectory $Y^0(k)$.
2. Linearization of the nonlinear model, involving n calculations of this model when applying numerical approximation of partial derivatives. Then calculation of the dynamic matrix $\mathbf{M}(k)$ of dimension $(N \cdot n) \times (N_u \cdot n)$, according to (21).

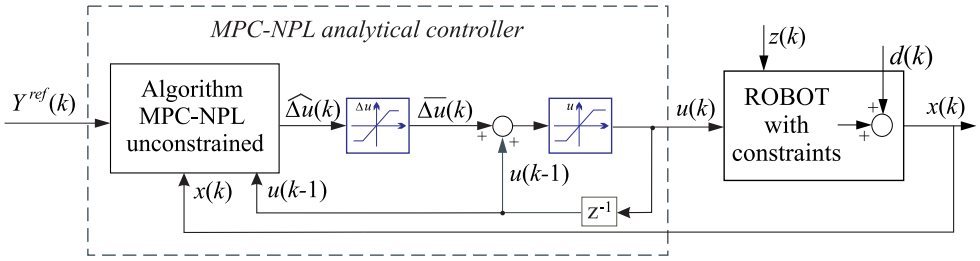

Figure 1. Structure of the MPC-NPL analytical algorithm

3. Solving the MPC optimization problem. If numerical algorithm is applied, then QP procedure is used and the computational load depends mainly on the number of optimization variables which is $N_u \cdot n$. When applying analytical algorithm, the set of $N_u \cdot n$ linear equations (24) is solved or substitution (28) is calculated, with computational load due to computation (and possibly inversion) of quadratic matrix $\mathbf{M}(k)^T \underline{\Psi} \mathbf{M}(k) + \underline{\Lambda}$ of dimension $N_u \cdot n$.

Looking at the above analysis, it can be seen that the computational load depends on manipulator dimensionality n, what could be expected. It depends also on the length of prediction and control horizons, N and N_u, respectively, which are design parameters of MPC algorithms. To decrease the time of computations as much as possible, the horizons should be as short as possible – but preserving adequate control performance. This concerns first of all control horizon N_u, as it directly influences dimensionality of the MPC optimization problem. Luckily, N_u can be very short – in the example manipulator considered in the next section, $N_u = 2$ occurred to be a very good choice, even $N_u = 1$ could have been considered. The length of prediction horizon N must be sufficiently long, to assure appropriate control performance and robustness, see, e.g., [9, 34] – but no longer than necessary (theoretically, it is not limited from above). It should be also noted that length of N depends on physical dynamics of the controlled process (the manipulator), thus the faster sampling of the MPC controller, the longer N must be designed.

Having selected the horizons, organization of calculations is crucial to decrease the time. First, the matrix (21) can be calculated efficiently, despite its complexity. It has Toeplitz block-structure; thus, only its first block-column consisting of matrices $\mathbf{M}_i(k)$ must be calculated. Due to structure of eq. (22), this calculations can be effectively organized in a recursive way:

$$\begin{aligned} \mathbf{P}_1 &= \mathbf{B}(k), & \mathbf{M}_1(k) &= \mathbf{C}\mathbf{P}_1, \\ \mathbf{P}_i &= \mathbf{B}(k) + \mathbf{A}(k)\mathbf{P}_{i-1}, & \mathbf{M}_i(k) &= \mathbf{C}\mathbf{P}_i, \; i = 2, \ldots, N \end{aligned} \tag{29}$$

thus consisting of $N-1$ multiplications and additions of minor size matrices, as multiplication by $\mathbf{C}$ means reduction of each $\mathbf{P}_i$ to its first n rows. Certainly, computational load depends also on complexity of the manipulator itself, on its number of degrees of freedom n, which is also the number of nonlinear equations in the model used to calculate the initial trajectory in point 1 and linearized model in point 2 above. Thus, dimension of the matrix $\mathbf{M}(k)$ (21), dependent on $\mathbf{A}(k)$, $\mathbf{B}(k)$ and $\mathbf{C}$ of the linearized model, also depends on n.

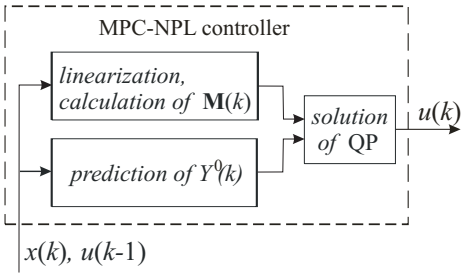

Figure 2. Parallel computational structure of MPC-NPL

It should be noted that calculations enlisted in points 1 and 2 above can be performed independently, the only common element are the initial data. Thus, applying two-processor parallel computation structure should lead to reduction of computation time. Such structure is depicted in Figure 2.

Expierience in application of MPC-NPL algorithms shows that linearization may be updated less frequently, not at every sampling instant, but repeated every N_{rlin} samples only, $N_{rlin} > 1$. In particular, this works well for weaker nonlinearities or when fast sampling is applied, which is standard in manipulator control. Then, control structure can be also designed with linearization and computation of $\mathbf{M}(k)$ performed by a distinct supervisory processor (computer), whereas basic processor calculates the initial output trajectory and performs the optimization, within the short sampling period T_c.

Linearization is the same both in the numerical and analytical NPL algorithms. But there is a difference in optimization problems. In the numerical algorithm, the supervisory processor calculates and transmits matrix $\mathbf{M}(k)$ to the basic processor, needed to formulate the QP problem. In the the analytical algorithm, more calculations can be shifted to the supervisory processor. As the same matrix $\mathbf{M}(k)$ is used during subsequent N_{rlin} sampling periods, it is better to use explicit formula (28) than to solve linear equations (24).

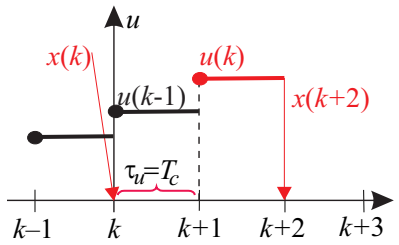

Figure 3. Time dependences in digital control system with unitary computational control delay τ_u

Moreover, what the basic processor really needs is the first n rows of the matrix

$$\mathbf{K}(k) = \left[\mathbf{M}(k)^T \underline{\Psi} \mathbf{M}(k) + \underline{\Lambda}\right]^{-1} \mathbf{M}(k)^T \underline{\Psi}, \qquad (30)$$

calculated by the supervisory processor. Denote by $\mathbf{K}_1(k)$ first n rows of $\mathbf{K}(k)$, then optimization task performed by the basic processor reduces to substitution

$$\widehat{\Delta u}(k) = \mathbf{K}_1(k)[Y^{ref}(k) - Y^0(k)], \qquad (31)$$

followed by trimming $\widehat{\Delta u}(k)$ to the constraints, as described in the previous section. This is the fastest realization of the MPC-NPL algorithm.

Despite efforts to minimize the time of computations as much as possible, this time may be comparable or almost equal to the length of the sampling period T_c, for fast sampling. Thus, it introduces computational control delay τ_u to the feedback loop. The most practical case it that of $\tau_u = T_c$, which will be further considered. Possibility to use the model augmented by this delay is an advantage of the MPC algorithms. For the control delay $\tau_u = T_c$, this results in augmenting the manipulator state to $x^T = [q^T \; \dot{q}^T \; z^T] \in R^{3n}$, and the state equations (3) to

$$x(k+1) = \begin{bmatrix} q(k) + T_c\, \dot{q}(k) \\ \dot{q}(k) - T_c\, M(q(k))^{-1}[C(x(k))\dot{q}(k) + \\ u(k) \\ \qquad + F(\dot{q}(k)) + g(q(k)) - z(k)\,] \end{bmatrix} \qquad (32)$$

It should be pointed out that the mentioned computational delay τ_u is an additional unitary delay, which adds to the standard unitary delay of digital control – resulting in the first reaction of the state on the control signal $u(k)$ after two sampling periods. This is schematically shown in Figure 3, where $u(k)$ denotes control signal calculated using information (measurements) obtained at time instant k.

3. Control of a direct drive manipulator

3.1. Models for Manipulator Simulation and Controller Design

The planar two-link experimental direct drive manipulator (EDDM) will be considered, it is described in detail in [35]. Its schematic diagram is shown in Figure 4. Vectors of joint angles

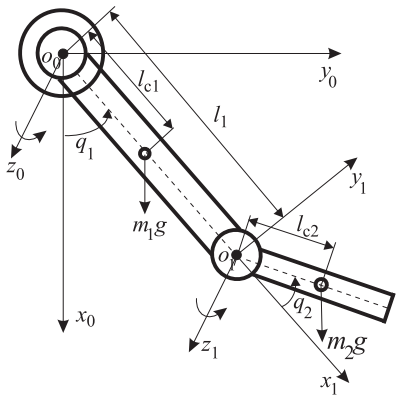

Figure 4. Schematic diagram of the experimental direct drive manipulator

and control torques are $q(t) = [q_1(t) \;\; q_2(t)]^T$, $u(t) = [u_1(t) \;\; u_2(t)]^T$, respectively. Matrices and function of its model (1) are:

$$M(q) = \begin{bmatrix} p_1 + 2p_3\cos(q_2) & p_2 + p_3\cos(q_2) \\ p_2 + p_3\cos(q_2) & p_2 \end{bmatrix}, \qquad (33)$$

$$C(q,\dot{q}) = \begin{bmatrix} -p_3\sin(q_2)\dot{q}_2 & -p_3\sin(q_2)(\dot{q}_1 + \dot{q}_2) \\ -p_3\sin(q_2)\dot{q}_1 & 0 \end{bmatrix}, \qquad (34)$$

$$F(\dot{q}) = \begin{bmatrix} f_{v1} & 0 \\ 0 & f_{v2} \end{bmatrix}\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} f_{C1} & 0 \\ 0 & f_{C2} \end{bmatrix}\begin{bmatrix} \mathrm{sgn}(\dot{q}_1) \\ \mathrm{sgn}(\dot{q}_2) \end{bmatrix}$$
$$= F_v\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + F_C\begin{bmatrix} \mathrm{sgn}(\dot{q}_1) \\ \mathrm{sgn}(\dot{q}_2) \end{bmatrix}, \qquad (35)$$

$$g(q) = \begin{bmatrix} p_4\sin(q_1) + p_5\sin(q_1 + q_2) \\ p_5\sin(q_1 + q_2) \end{bmatrix}. \qquad (36)$$

Physical data given in [35] lead to the following values of model parameters:

$p_1 = 2.352\,Nms^2$, $p_2 = 0.102\,Nms^2$,
$p_3 = 0.084\,Nms^2$,
$p_4 = 38.466\,Nm$, $p_5 = 1.825\,Nm$
$f_{v1} = 2.288\,Nms$, $f_{v2} = 0.175\,Nms$,
$f_{C1} = 7.17\,Nm$ for $\dot{q}_1 > 0$,
$f_{C1} = 8.05\,Nm$ for $\dot{q}_1 < 0$,
$f_{C2} = 1.734\,Nm$,

together with limits on control torques:

τ_{1max}=200 Nm, τ_{2max}=15 Nm.

We used the discrete-time version (3)-(4) of the model for simulation of the manipulator itself within the feedback control loop, with very short integration step $T_p = 0.0001s$. Its discrete-time dynamics mimics then perfectly the continuous-time one. For design of MPC controllers, a simplified continuous and differentiable model was used, with Coulomb friction omitted (treated as unmodeled dynamics), i.e. $F(\dot{q}) = F_v\dot{q}$. The controller sampling period T_c ($T_c \gg T_p$) was used.

Table 1. Values of ISE for MPC algorithms with cubic reference trajectories, for few N_u and different disturbances

disturbances	N_u	MPC-NO	MPC-NPLn	NPL-NPLa
Cf (Coulomb friction)	1	$0.1120 \cdot 10^{-3}$	$0.1122 \cdot 10^{-3}$	$0.1122 \cdot 10^{-3}$
	2	$0.0321 \cdot 10^{-3}$	$0.0321 \cdot 10^{-3}$	$0.0321 \cdot 10^{-3}$
	3	$0.0234 \cdot 10^{-3}$	$0.0234 \cdot 10^{-3}$	$0.0234 \cdot 10^{-3}$
	4	$0.0230 \cdot 10^{-3}$	$0.0230 \cdot 10^{-3}$	$0.0230 \cdot 10^{-3}$
Cf + external (Zu)	1	$0.1120 \cdot 10^{-3}$	$0.1122 \cdot 10^{-3}$	$0.1122 \cdot 10^{-3}$
	2	$0.0339 \cdot 10^{-3}$	$0.0339 \cdot 10^{-3}$	$0.0339 \cdot 10^{-3}$
	3	$0.0250 \cdot 10^{-3}$	$0.0250 \cdot 10^{-3}$	$0.0250 \cdot 10^{-3}$
Cf + parametric (P1)	1	$0.0983 \cdot 10^{-3}$	$0.0983 \cdot 10^{-3}$	$0.0983 \cdot 10^{-3}$
	2	$0.0310 \cdot 10^{-3}$	$0.0310 \cdot 10^{-3}$	$0.0310 \cdot 10^{-3}$
	3	$0.0252 \cdot 10^{-3}$	$0.0252 \cdot 10^{-3}$	$0.0252 \cdot 10^{-3}$

The following scenario of simulations was applied:

- Simulation horizon: length 2.5 s. Reference trajectories with two changes of joints positions (angular, in radians), first change starting at 0.15 s, second at 1.5 s. The changes are implemented:
  1) along cubic trajectories, with each cubic trajectory duration time 0.5 s.
  2) as steps, i.e., reference trajectories are piecewise constant.
- Disturbances:
  - external (Z1): a torque $4\,Nm$ added at the input to the second link at middle point 1.25 s of the simulation horizon; a disturbance typical for testing controller performance, including manipulator controllers, see, e.g., [3]; or
  - parametric uncertainty (P1): change od manipulator dynamics by adding a mass of 1 kg at the end of the second arm. This leads to change of manipulator parameters to the values:
    $p_1 = 2.596\,Nms^2$, $p_2 = 0.144\,Nms^2$,
    $p_3 = 0.154\,Nms^2$,
    $p_4 = 42.88\,Nm$, $p_5 = 3.351\,Nm$.
    These parameters define the manipulator as the process in the feedback loop only, controllers are designed using the previously given nominal values of parameters.
- Control performance defined by ISE (integrated squared errors) criterion:

$$\text{ISE} = \sum_{k=1}^{N_{sym}} \left(e_1(k)^2 + e_2(k)^2\right) * T_c, \qquad (37)$$

where N_{sym} is the number of sampling periods T_c along the simulation horizon.

It should be pointed out that the assumed disturbance and parameter inaccuracy are quite significant. The disturbing torque is about 27% of maximal second actuator torque ($15\,Nm$), changes in parameters $p_1, \ldots, p_5$ are also quite large. Additionally, nonlinear unmodelled dynamics is present (Coulomb friction).

In the design process of MPC controllers, weighting parameters in the performance function has been found to be appropriate with values $[\psi_1\ \psi_2] = [10\ 10]$, $[\lambda_1\ \lambda_2] = [0.0001\ 0.001]$, assuring stable and robust performance. Notice that differences in values of the weights λ and ψ correspond to differences in ranges of the corresponding variables (scaling). Sampling period T_c = 0.005 s was basically assumed, but performance with another values was also investigated. Different prediction horizons were tested, N=18 has been chosen for T_c = 0.005 s as a tradeoff between computational load and performance/robustness.

3.2. Results with MPC-NO and MPC-NPL Controllers

A comprehensive comparison of control performance with all MPC algorithms presented in this paper is given in Table 1, for the case with cubic reference trajectories. It is in terms of values of ISE index (37), which makes the comparison easier. Simulations were performed in Matlab environment, using for nonlinear optimization "fmincon" and for QP "quadprog" procedures. For model linearization, "Symbolic Math" toolbox was tried. But, due to extremely complex formulae obtained, numerical approximation of partial derivatives was applied, with results practically equivalent, very small differences did not influenced control system behaviour. Three cases were considered: with unmodeled dynamics denoted by "Cf" (Coulomb friction) only, with Cf and external disturbing torque (Zu) or parameter uncertainty (P1) added.

Very short control horizons N_u=1 and N_u=2 assure the smallest dimensionality of nonlinear optimization problems, $N_u \times n = 2$ and 4, respectively, thus the shortest online computation (optimization) time. The choice N_u= 2 results in better control performance, but performance with N_u= 1 may be acceptable when computation time is critical. ISE value with N_u= 3 is only slightly better than for N_u= 2, but with increased computational load. For each of the disturbance cases, results for all three algorithms, NO, NPLn and NPLa, are practically identical. The trajectories are visually undistinguishable, therefore are presented for the MPC-NPLn algorithm only, in Figures 5 and 6, for external disturbance and parameter uncertainty, respectively. Notice that external disturbance and parameter uncertainty are well attenuated, with minor differences for the same control horizons.

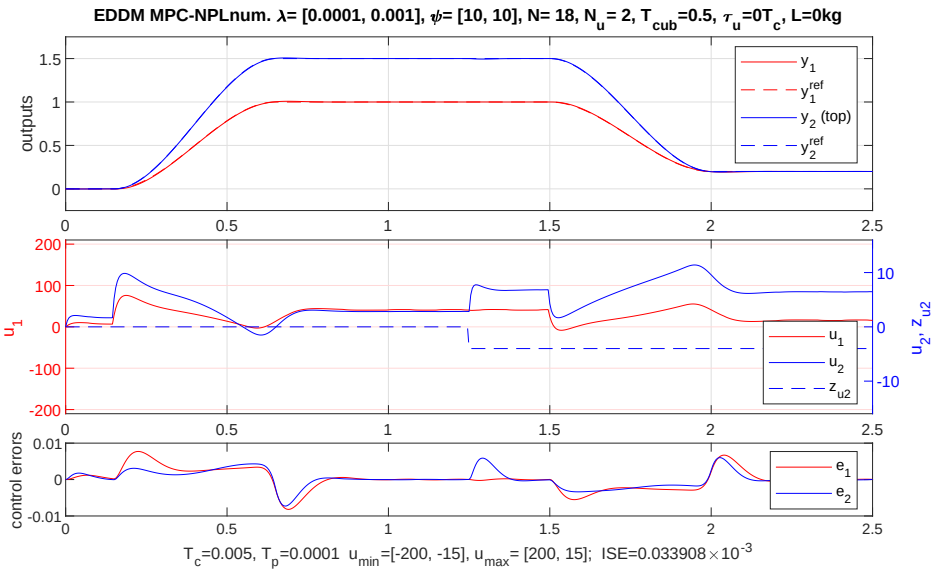

Figure 5. Trajectories for MPC-NPLn with N_u=2 and external disturbance (Zu)

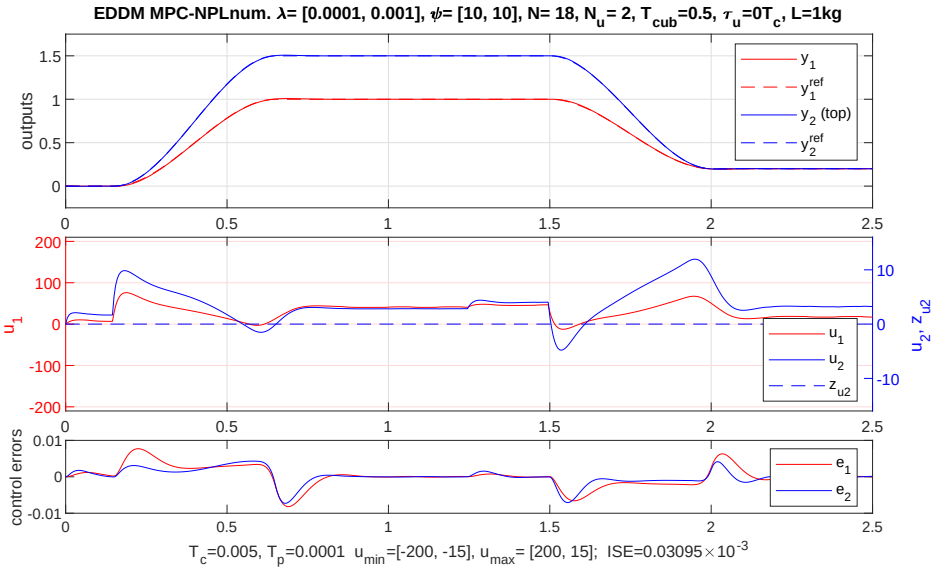

Figure 6. Trajectories for MPC-NPLn with N_u=2 and parametric disturbance (P1)

For piecewise constant reference trajectories, a direct application of even the MPC algorithms (as presented hitherto) leads usually to excessive saturation of control signals and to overshoots. But it is possible to avoid this, by adding internal MPC mechanism transforming piecewise constant reference trajectories $y^{ref}(k)$ into internal reference trajectories $y^{iref}(k)$ of exponential type, see, e.g., [17], [27]. This involves replacing the first sum in the performance function $J(k)$ in (7) with the sum

$$J_1(k) = \sum_{p=1}^{N} \left\| y^{iref}(k+p|k) - y(k+p|k) \right\|_{\Psi}^2, \quad (38)$$

where

$$y^{iref}(k+p|k) = [1-(T_{ref})^p] y^{ref}(k) + (T_{ref})^p y(k), \quad p = 1, \ldots, N \quad (39)$$

and $0 \leq T_{ref} < 1$ is a scaling parameter. For $T_{ref} = 0$, $y^{iref}(k+p|k) = y^{ref}(k)$, but the larger $T_{ref} > 0$ the slower the convergence of $y^{iref}(k+p|k)$ to $y^{ref}(k)$, over the prediction horizon. This technique enables to achieve smooth, overshoot-free trajectories of the manipulator arms.

Figures 7 and 8 show manipulator trajectories for MPC-NPLn algorithm with and without internal reference trajectory technique, respectively, with external disturbance. The advantage of using this technique can be clearly seen. Comparison of Figures 5 and 7 leads to conclusion that the approach described now is a possible alternative to smooth cubic transitions between positions, but in the cubic case the trajectories of control inputs are more smooth, without saturation.

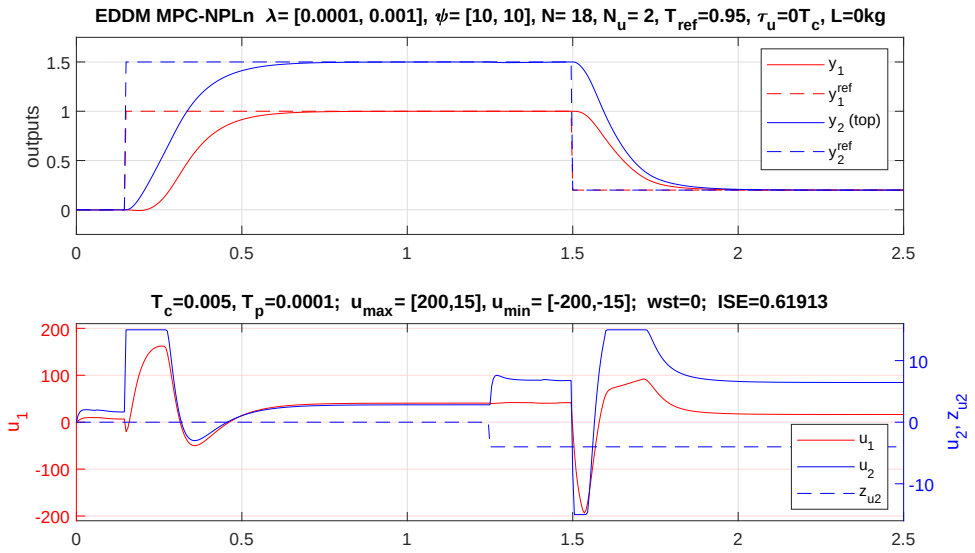

Figure 7. Trajectories for MPC-NPLn with N_u=2, $T_{ref} = 0.95$ and disturbance Zu; piecewise constant reference trajectories

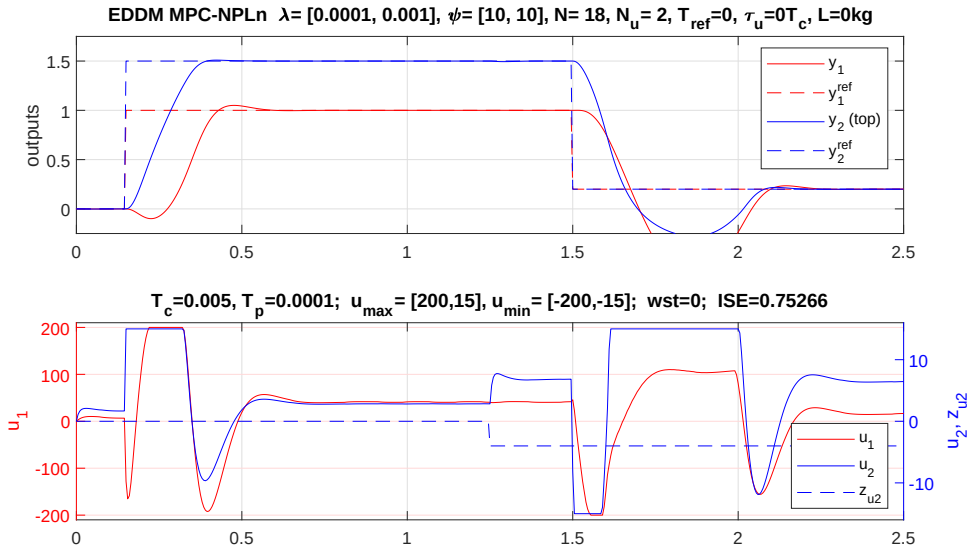

Figure 8. Trajectories for MPC-NPLn with N_u=2, $T_{ref} = 0$ and disturbance Zu; piecewise constant reference trajectories

More comprehensive results are shown in Table 2, in terms of ISE values. Trajectories corresponding to these results are similar to those shown in Figure 7. The situation is with significant constraint activity (saturation of control inputs), therefore application of analytical algorithm leads to inferior results.

The presented results clearly indicate, for both smooth cubic and abrupt step changes between reference positions, that much simpler MPC-NPL algorithms can be used practically without loss of control quality, instead of more computationally involved and less computationally robust MPC-NO one (as nonlinear optimization is less robust than a quadratic one). The loss of performance can be only awaited with most simple MPC-NPL analytical version, in cases with significant activity of constraints.

3.3. CTC-PID and CTC-PID2dof Control, Comparisons with MPC

To evaluate performance, applicability of MPC algorithms to control of robotic manipulators fairly, a comparison with existing technique should be done. The most commonly used in industry is the classical decentralized PID control, e.g., [1, 4, 36], particularly when joint motors are with gears, reducing cross-coupling effects between joints. However, implementation and tuning of PID controllers must be made with care, to overcome possible overshoots.

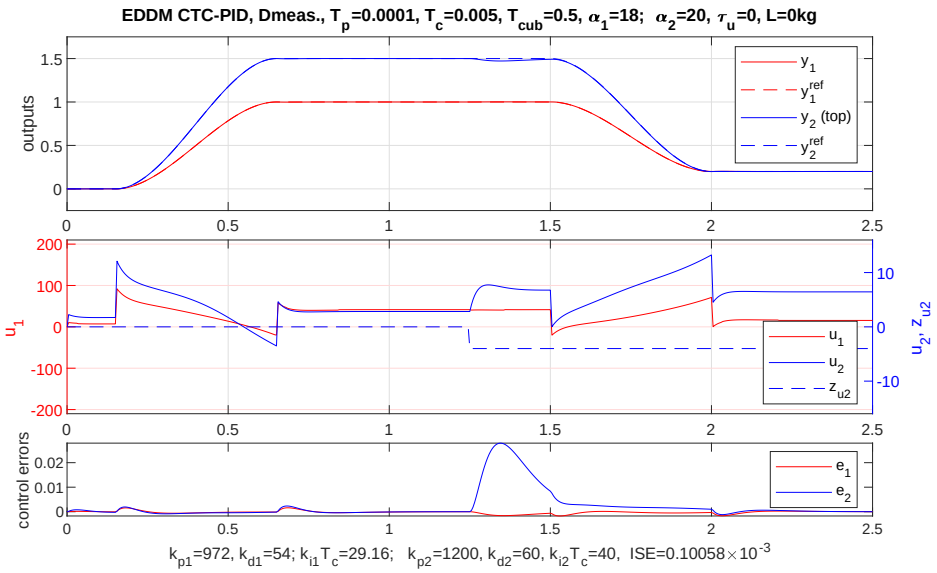

Figure 9. CTC-PID control, with external step disturbance at 1.25s

Table 2. ISE for MPC algorithms with T_{ref}=0.95, for two values of N_u and different disturbances, for piecewise constant reference trajectories;
[1)] overshoot 4% for second step, eliminated with N=22,
[2)] overshoot 18% for second step, with N=22 reduced to 10%

disturb.	N_u	MPC-NO	MPC-NPLn	MPC-NPLa
Cf	1	0.6435	0.6454	0.7508
	2	0.6339	0.6343	0.7291
Cf+Zu	1	0.6309	0.6343	0.7392
	2	0.6180	0.6191	0.7092
Cf+P1	1	0.6699	0.6725	0.7871
	2	0.6627 [1)]	0.6645 [1)]	0.7621 [2)]

Possible solutions are, e.g., partitioned PD control or PID with reference trajectory prefiltering, see, e.g., [4]. In some instances, double-loop P-PI control of position and velocity can be utilized. However, for manipulators with stronger interactions, like those with direct drive motors, the model-based algorithms are rather recommended. The well known solution is here the computed torque control (CTC), which linearizes and decouples the manipulator, with PD/PID controllers with acceleration feedforward on top of the linearized manipulator, see, e.g., [1, 4, 36]. Comparison of MPC with this control structure seems to be fair, as both are nonlinear and model-based.

Recall the CTC-PID controller algorithm, see, e.g., [1, 4, 36]:

$$u = M(q)(\ddot{q}^{ref} + u_{PID}) + C(q,\dot{q})\dot{q} + F(\dot{q}) + g(q), \quad (40)$$

where u_{PID} is the output of a multiloop PID controller. Inserting (40) into the manipulator dynamics (1) results in closed loop dynamics consisting of n third order dynamical systems with PID gains k_{p_j}, k_{d_j} and k_{i_j} as parameters, $j = 1, ..., n$. Assuming pole placementn design, poles of characteristic equations yields formulae for PID gains.

Table 3. Values of ISE for CTC-PID and MPC-NPL algorithms, for different disturbances, with cubic reference trajectories between positions

CTC-PID (α_1=18, α_2=20)		MPC-NPLn (N=18, N_u=2)	
disturb.	ISE	disturb.	ISE
none	$0.0015 \cdot 10^{-3}$	Cf	$0.0321 \cdot 10^{-3}$
Zu	$0.1006 \cdot 10^{-3}$	Cf+Zu	$0.0339 \cdot 10^{-3}$
P1	$0.1485 \cdot 10^{-3}$	Cf+P1	$0.0310 \cdot 10^{-3}$

A popular choice to get overshoot-free response is to assume one triple real pole, see [4, 37], then tuning all gains of every SISO PID controller is by one parameter, $k_{p_j} = 3\alpha_j^2$, $k_{d_j} = 3\alpha_j$, $k_{i_j} = \alpha_j^3$, where $-\alpha_j$ ($\alpha_j > 0$) is the assumed pole value. This value is usually chosen according to required settling time t_s, for a triple pole $\alpha \approx 8/t_s$, see [37]. Assuming $t_s = 0.5$ s for both arms (time equal to the period of position change along cubic trajectories), we get $\alpha_1 = \alpha_2 = 16$. However, for tracking smooth trajectories like cubic ones further fine-tuning is reasonable, by trial and error method. After that we have finally chosen $\alpha_1 = 18$, $\alpha_2 = 20$.

Selected simulation results are presented in Table 3 and in Figure 9. For perfect modeling (without disturbances), CTC-PID is superior. But under disturbances, MPC-NPL provides better performance: significantly better ISE values, smaller control errors and, moreover, more smooth control trajectories, as can be seen comparing Figures 9 and 5. Despite the fact that Coulomb friction is present in CTC feedback model (40), see (35), whereas it is treated as umodeled dynamics in MPC algorithms.

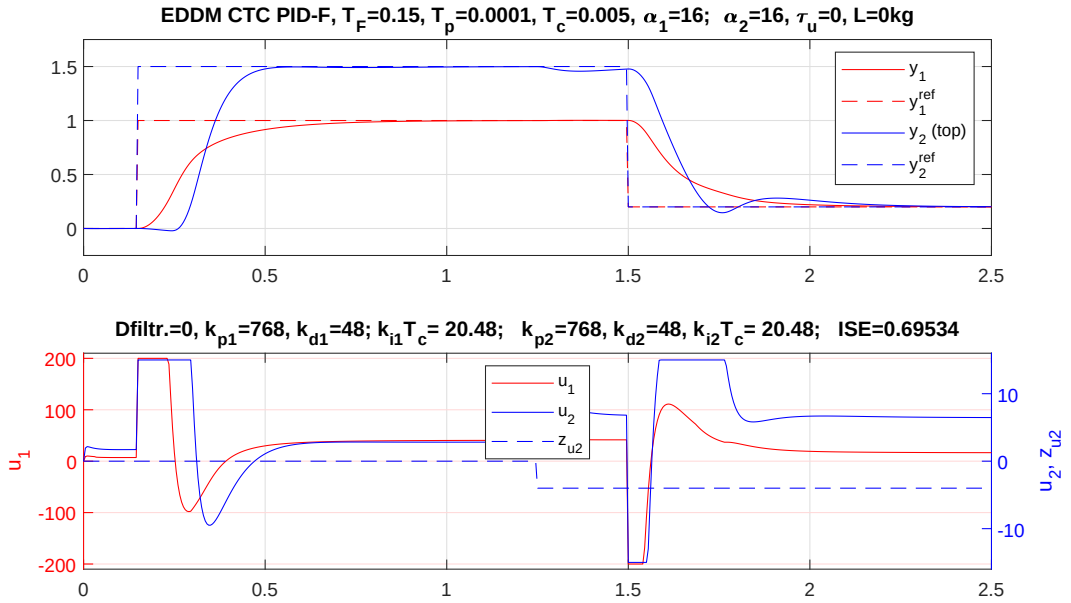

Figure 10. CTC-PID2dof (CTC PID-F) control, T_F=0.15s, with external step disturbance at at 1.25s

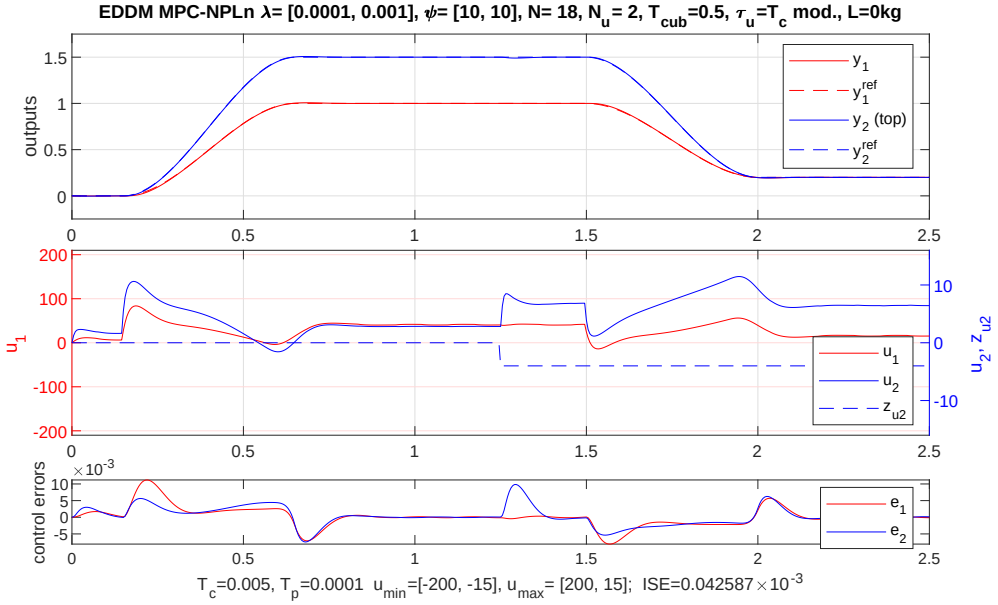

Figure 11. Trajectories for MPC-NPLn with T_c=0.005s and control delay $\tau_u = T_c$, external disturbance Zu

Table 4. Values of ISE and overshoot κ for CTC-PID2dof and MPC-NPL algorithms, for different disturbances, with piecewise constant reference trajectories

CTC-PID2dof (α_1=α_2=16, T_F=0.15)			**MPC-NPLn** (N=18, N_u=2)		
disturb.	**ISE**	κ	**disturb.**	**ISE**	κ
none	0.7221	0%	Cf	0.6343	0%
Zu	0.6953	6%	Cf+Zu	0.6191	0%
P1	0.7731	39%	Cf+P1	0.6645	4%

It is generally not recommended to make abrupt step changes of constant reference positions, when leaving to shape manipulator transition trajectories to dynamics of the feedback control loop with standard PID controllers only, as this usually leads to excessive saturation of control signals and large overshoots. However, augmenting PID controllers by dynamic prefiltering of piecewise constant reference signals improves the situation. Such PID structure with a set-point prefilter is usually called a PID 2dof controller. We applied such structure to our manipulator, with first order inertial prefilters with time constant T_F = 0.15s, for each arm. For PID settings, we applied α_1 = α_2 = 16, resulting from assumed settling time t_s = 0.5 s for both arms, see [37], to perfectly coincide with time of cubic position change. Adding prefilters to CTC-PID algorithms can be treated as a technique corresponding to adding internal reference trajectories to MPC algorithms and should, therefore, lead to a fair comparison of both algorithms. Selected results are presented in Table 4 and Figure 10.

Table 5. Values of ISE for MPC-NPL and CTC-PID algorithms for various sampling intervals T_c, without and with computational control delay τ_u=T_c, for cubic transition trajectories between reference positions

T_c	τ_u	MPC-NPL		CTC-PID	
		disturbances	ISE	disturbances	ISE
0.01	0	Cf+Zu	$0.1010 \cdot 10^{-3}$	Zu	$0.1742 \cdot 10^{-3}$
	T_c	Cf+Zu	$0.1525 \cdot 10^{-3}$	Zu+τ_u	$0.2278 \cdot 10^{-3}$
	0	Cf+P1	$0.1033 \cdot 10^{-3}$	P1	$0.1561 \cdot 10^{-3}$
	T_c	Cf+P1	$0.1676 \cdot 10^{-3}$	P1+τ_u	$0.2374 \cdot 10^{-3}$
0.005	0	Cf+Zu	$0.0339 \cdot 10^{-3}$	Zu	$0.1006 \cdot 10^{-3}$
	T_c	Cf+Zu	$0.0426 \cdot 10^{-3}$	Zu+τ_u	$0.1074 \cdot 10^{-3}$
	0	Cf+P1	$0.0310 \cdot 10^{-3}$	P1	$0.1485 \cdot 10^{-3}$
	T_c	Cf+P1	$0.0430 \cdot 10^{-3}$	P1+τ_u	$0.1651 \cdot 10^{-3}$
0.0025	0	Cf+Zu	$0.0113 \cdot 10^{-3}$	Zu	$0.1004 \cdot 10^{-3}$
	T_c	Cf+Zu	$0.0127 \cdot 10^{-3}$	Zu+τ_u	$0.1021 \cdot 10^{-3}$
	0	Cf+P1	$0.0103 \cdot 10^{-3}$	P1	$0.1467 \cdot 10^{-3}$
	T_c	Cf+P1	$0.0122 \cdot 10^{-3}$	P1+τ_u	$0.1521 \cdot 10^{-3}$

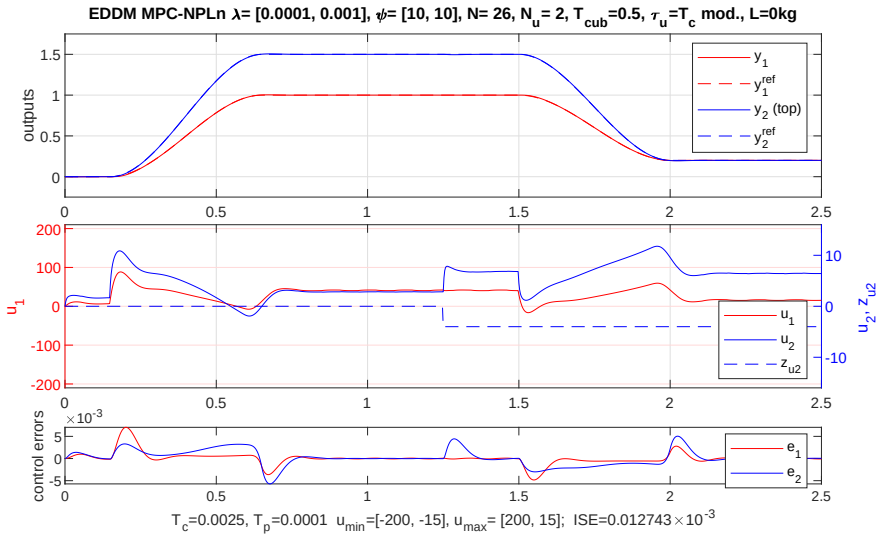

Figure 12. Trajectories for MPC-NPLn with T_c=0.0025s and control delay $\tau_u = T_c$, external disturbance Zu

The results indicate that the MPC algorithm provides better performance, especially under disturbances. Despite prefiltering, the CTC-PID2dof (CTC-PID-F) algorithm suffers then overshoots and slightly longer settling time (overshoots would be much larger without prefiltering). When these defficiences are acceptable, this algorithm could be a design alternative.

3.4. Impact of Sampling Period and Computation Time Delay

Model-based algorithms need to perform more computations than simpler PID algorithms. This can result in tangible delay in implementation of control signals due to computation time.

Therefore, we investigated impact of both sampling period length and control delay on performance of algorithms considered in the paper, in particular MPC-NPL and CTC-PID. Most practically important is the delay equal to one sampling period. It means that, using new measurements obtained at the beginning of the sampling period, calculations of the new controller output must be finished within this period, to send this output to the actuators at the end of this period (or beginning of the next one).

The influence of such additional unitary computational time delay $\tau_u = T_c$ on control performance was investigated. Selected results of comparisons are summarized in Table 5, for both cases of considered disturbances. The cases without control delays are also given, for easy comparisons. Control with three sampling periods was investigated: T_c= 0.01 s, T_c= 0.005 s and T_c= 0.0025 s. In MPC algorithms, appropriate relation of the prediction horizon N to the process dynamics must be preserved, physically.

That is why longer prediction horizons N, given in numbers of sampling periods, correspond to shorter values of T_c. Values of N equal to 12, 18 and 26 were chosen for decreasing values of T_c, respectively. But N_u=2 could be used in all cases as effective control horizon.

Looking at the results for MPC, it can be seen that decreasing T_c improves ISE values. Decrease by factor 2, from 0.01 s to 0.005 s, decreases ISE even more, the same occurred for the decrease from 0.005 s to 0.0025 s. Considering results without and with control delay, we can see that the smaller T_c the smaller difference between the corresponding values. For largest value T_c=0.01, deterioration of ISE after introduction of control delay is most visible. That is the reason why T_c=0.005s has been chosen as the basic sampling period in the paper. With decrease of ISE, control errors should decrease similarly, exemplary confirmation is given by comparing Figures 11 and 12, showing manipulator trajectories with control delay for T_c= 0.005 s and 0.0025 s.

Analysing results obtained with CTC-PID control we can see that increase of T_c from 0.005 s to 0.01 s increases ISE values significantly, but with decrease of T_c to 0.0025 s improvement is marginal. On the other hand, with MPC this improvement was significant. The reason is that for fast sampling PID tuning is as for continuous-time control. Therefore, the faster sampling the better discrete-time PID controller mimics continuous-time PID one and the smaller differences when decreasing the sampling period. For obvious reasons, the smaller T_c the less the negative impact of computational delay.

4. Conclusion

Application of effective and most up-to-date nonlinear MPC algorithms with state-space models to manipulator control was presented, in comparison with standard and enhanced realizations of CTC-PID algorithms. MPC algorithms were presented with most effective, recently proposed disturbance attenuation technique [30], which avoids necessity of dynamic modeling of disturbances, both external or internal, or to resort to alternative additional techniques to attenuate disturbances, like SMC, as met in several papers on MPC control of manipulators.

The core of the paper is computationally effective MPC-NPL algorithm (Nonlinear Prediction with Linearization), presented in two versions: the first with constrained QP optimization and the second, computationally simpler, with unconstrained optimization and a posteriori matching the unconstrained result with inequality constraints. Detailed discussion on organization of calculations leading to shortest execution time was provided for MPC-NPL algorithms. For all algorithms a comprehensive comparative simulation study was presented in the paper, with experimental direct drive manipulator, under significant external and parametric disturbances. Results were compared with those obtained with known CTC-PID algorithm, which is also model-based and using the nonlinear manipulator model. The comparisons were performed for two shapes of reference trajectories: smooth cubic or piecewise constant. For the second case, MPC with additional internal reference trajectories and CTC-PID2dof algorithm have been proposed and shown to lead to practically useful results, which seems to be a novelty in manipulator control literature. For all considered cases, the MPC-NPL algorithms have been shown to perform better or significantly better than CTC-PID algorithms. However, when certain deterioration in accuracy and small overshoots (which can occur after step changes of reference trajectories) are acceptable, CTC-PID or CTC-PID2dof algorithms could be a design alternative.

Additional contribution of the paper is investigation of the influence of sampling period length and computational delay on performance of MPC and CTC-PID algorithms. This is important for design of model-based control algorithms with fast sampling, as they require more computations.

It is believed that the results presented in the paper should be interesting for both applied researchers and industrial practitioners in the field of manipulator control.

AUTHOR

Piotr Tatjewski* – Warsaw University of Technology, Nowowiejska 15/19, Warsaw, Poland, e-mail: piotr.tatjewski@pw.edu.pl.

*Corresponding author

References

[1] M.W. Spong. An historical perspective on the control of robotic manipulators. *Annual Reviews of Control, Robotics, and Autonomous Systems*, 5:1–31, 2022.

[2] M.F. Khan, R. Islam, and J. Iqbal. Control strategies for robotic manipulators. In *Proceedings of the 2012 International Conference of Robotics and Artificial Intelligence*, Rawalpindi, Pakistan, 2012.

[3] F.L. Lewis, D.M. Dawson, and C.T. Abdallah. *Robot Manipulator Control Theory and Practice*. CRC Press, Boca Raton, 2003.

[4] R. Kelly, V. Santibáñez, and A. Loría. *Control of Robot Manipulators in Joint Space*. Springer, London, 2005.

[5] E.F. Camacho and C. Bordons. *Model Predictive Control*. Springer Verlag, London, 1999.

[6] J.M. Maciejowski. *Predictive Control*. Prentice Hall, Harlow, England, 2002.

[7] T. L. Blevins, G. K. McMillan, W. K. Wojsznis, and M. W. Brown. *Advanced Control Unleashed*. The ISA Society, Research Triangle Park, NC, 2003.

[8] S.J. Qin and T.A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11:733–764, 2003.

[9] P. Tatjewski. *Advanced Control of Industrial Processes*. Springer Verlag, London, 2007.

[10] L. Wang. *Model Predictive Control System Design and Implementation using MATLAB*. Springer Verlag, London, 2009.

[11] K.J. Holkar and L.M. Waghmare. An overview of model predictive control. *International Journal of Control and Automation*, 3(4):47–63, 2010.

[12] T. L. Blevins, W. K. Wojsznis, and M. Nixon. *Advanced Control Foundation*. The ISA Society, Research Triangle Park, NC, 2013.

[13] J.B. Rawlings, D.Q. Mayne, and M.M. Diehl. *Model Predictive Control: Theory, Computation, and Design 2nd Edition*. Nob Hill Publishing, Santa Barbara, California, 2017.

[14] I.L. Huang, H.H. Lou, J.P. Gong, and T.F. Edgar. Fuzzy model predictive control. *IEEE Transactions on Fuzzy Systems*, 8(6):665–678, 2000.

[15] M. Ławryńczuk. *Computationally Efficient Model Predictive Control Algorithms: A Neural Network Approach. Studies in Systems, Decision and Control, Vol. 3*. Springer Verlag, Heidelberg, 2014.

[16] M.M. Morato, J.E. Normey-Rico, and O. Sename. Model predictive control design for linear parameter varying systems: A survey. *Annual Reviews in Control*, 49:64–80, 2020.

[17] M. Ławryńczuk and P. Tatjewski. Offset-free state-space nonlinear predictive control for Wiener systems. *Information Sciences*, 511:127–151, 2020.

[18] T. Rybus, K. Seweryn, and J.Z. Sasiadek. Application of predictive control for manipulator mounted on a satellite. *Archives of Control Sciences*, 28(1):105–118, 2018.

[19] S. Kleff, A. Meduri, R. Budhiraja, N. Mansard, and L. Righetti. High-frequency nonlinear model predictive control of a manipulator. In *Proceedings of the IEEE International Conference on Robotics and Automation*, Xi'an, China, 2021.

[20] P. Bumroongsri and S. Kheawhom. Interpolation-based off-line MPC for LPV systems. In *Proceedings of the 10th IFAC International Symposium on Dynamics and Control of Process Systems*, Mumbai, India, 2013.

[21] P.S.G. Cisneros, A.Sridharan, and H. Werner. Constrained predictive control of a robotic manipulator using quasi-LPV representations. *IFAC Papers Online Conference Paper Archive*, 51(26):118–123, 2018.

[22] J. Wilson, M. Charest, and R. Dubay. Non-linear model predictive control schemes with application on a 2 link vertical robot manipulator. *Robotics and Computer-Integrated Manufacturing*, 41:23–30, 2016.

[23] A. Benniran. Predictive optimizing reference governor for constrained 2 dof's robot with abrupt set-point trajectories. *Journal of Applied Science, Sabratha University*, 1:39–49, 2018.

[24] A. Ferrara, G.P. Incremona, and L. Magni. A robust MPC/ISM hierarchical multi-loop control scheme for robot manipulators. In *Proceedings of the 52nd Conference on Decision and Control*, Florence, Italy, 2013.

[25] G.P. Incremona, A. Ferrara, and L. Magni. MPC for robot manipulators with integral sliding modes generation. *IEEE/ASME Transactions on Mechatronics*, 22(3):1299–1307, 2017.

[26] S. Bouzoualegh, E. Guechi, and Y. Zennir. Model predictive control of a three degrees of freedom manipulator robot. In *Proceedings of the 3rd International Conference on Advanced Systems and Emergent Technologies*, pages 84–89, Hammamet, Tunisia, 2019.

[27] D. Nicolis, F. Allevi, and P. Rocco. Operational space model predictive sliding mode control for redundant manipulators. *IEEE Transactions on Robotics*, 36(4):1348–1355, 2020.

[28] P. Tatjewski. Disturbance modeling and state estimation for offset-free predictive control with state-spaced process models. *International Journal of Applied Mathematics and Computer Science*, 24(2):313–323, 2014.

[29] P. Tatjewski. Offset-free nonlinear predictive control with measured state and unknown asymptotically constant disturbances. In K. Malinowski, J. Józefczyk, and J. Świątek, editors, *Aktualne problemy automatyki i robotyki*, pages 288–299. Academic Publisher EXIT, Warszawa, Poland, 2014.

[30] P. Tatjewski. Offset-free nonlinear Model Predictive Control with state-space process models. *Archives of Control Sciences*, 27(4):595–615, 2017.

[31] P. Tatjewski and M. Ławryńczuk. Algorithms with state estimation in linear and nonlinear model predictive control. *Computers and Chemical Engineering*, 143:1–19, 2000.

[32] P. Tatjewski. Nieliniowe sterowanie predykcyjne ramion manipulatorów (Nonlinear predictive control of manipulator arms). *Pomiary Automatyka Robotyka*, 27(2):47–58, 2023.

[33] L. Garcia and E. Rosero. Non-linear model-based predictive control for trajectory tracking and control effort minimization in a smartphone-based quadrotor. *Journal of Automation, Mobile Robotics and Intelligent Systems*, 16(4):13–18, 2022.

[34] P. Tatjewski. *Sterowanie zaawansowane procesów przemysłowych (Advanced Control of Industrial Processes), Second, revised edition* (e-book, in Polish). Academic Publishing House EXIT, Warszawa, 2016.

[35] F. Reyes and R. Kelly. Experimental evaluation of identification schemes on a direct drive robot. *Robotica*, 15:563–571, 1997.

[36] M. Spong, S. Hutchinson, and M. Vidyasagar. *Robot Modeling and Control*. J. Wiley and Sons, 2005.

[37] A. Bożek and L. Trybus. Tuning PID and PI-PI servo controllers by multiple pole placement. *Bulletin of the Polish Academy of Sciences Technical Sciences*, 70(1):1–12, 2022.